A DHT-based Lightweight Service Discovery Protocol for Mobile Ad Hoc Networks

Yifan Ge and Weiwei Sun

School of Computer Science and Technology,
Fudan Univesity, Shanghai, China
{geyifan,wwsun}@fudan.edu.cn

Abstract. Distributed Hash Table (DHT) has proven to be an efficient way of locating resources in peer-to-peer networks. As mobile ad hoc networks (MANETs) and peer-to-peer networks share similar properties such as self-configuring and decentralized, the service discovery in MANETs can also potentially benefit from the deployment of a DHT overlay. However, as DHTs are intended to be used in Internet based peer-to-peer networks, it is not appropriate to simply deploy such an overlay in the bandwidth sensitive MANETs.
In this paper, we propose a DHT-based lightweight service discovery protocol for MANETs, which efficiently implements a DHT overlay in MANETs with negligible control messages. The DHT substrate is introduced to help the locating of service providers and a clustering protocol is adopted to suppress the amount of control packets generated by service discovery. Our simulations conducted in the wireless network simulator also prove our protocol is swift, efficient and reliable as well.

1 Introduction

Service Oriented Architecture (SOA) is a set of software system design principles and methodologies, which separates the function of the system into distinct units, or services. SOA enhances the flexibility of the system, which is an essential requirement for the deployment on the communication networks.

Mobile Ad hoc Networks (MANETs) are wireless, multi-hop, self-configuring networks with no fixed infrastructure. In such a network, every mobile node acts as a router, forwarding the packets for other nodes to their destinations. Due to the infrastructure-less and decentralized features of this network model, MANETs enable ubiquitous computing and interoperability of the mobile nodes, where we can see a potential application scenario of the previously mentioned SOA.

The discoverability of services is one ground requirement of SOA that should be guaranteed. A service discovery protocol is used to allow the devices in the network to detect and locate the required services, and publish their own service capacities to the network. Although there are already a comprehensive set of mature service discovery protocols such as Web Services Dynamic Discovery (WS-Discovery)[1], Universal Description Discovery and Integration (UDDI)[2], etc., the service discovery in MANETs must be treated separately from the protocols designed to be used in the Internet.

A major reason makes service discovery protocol in MANETs so different from the protocols used in Internet is that the mobile nodes normally have very little or even no knowledge about the condition and capacity of each other, and this is caused by the nature of MANETs:

- Mobile nodes are free to move, and join or leave the network, which leads to a continuously changing network topology.
- The wireless channel condition can be unreliable as interference and collision frequently occur during the data transmission.
- Resources (processing capacity, memory, battery, etc.) are limited for the mobile nodes.

These characteristics constrain the utilization of service administrations or registration servers for service discovery in MANETs, and urge a decentralized and distributed manner to be adopted.

On the other hand, a peer-to-peer (P2P) overlay network is a network architecture allowing each computer in the network (peer) simultaneously functions as a client and a host to the other peers on the network. Such a network architecture works in a distributed manner, and as peers are equally privileged, central servers are not necessary to maintain the network architecture. The equality of the peers of a P2P overlay network resembles in large extents the behavior of mobile nodes in MANETs. Further similarities are also shared between these two types of networks:

- Both P2P overlay network and MANETs are self-configuring and decentralized. No central server is needed to coordinate the network.
- Peers and mobile nodes join and leave the network freely, which leads to a frequently changing topology of the two network models.
- The connection and communication between peers or mobile nodes are in a hop-by-hop manner. Although per-hop transmission on P2P overlay network is normally via TCP links anFslpd corresponds to several hops on underlying physical network, while per-hop transmission on MANETs are direct and via wireless UDP links.

The first P2P application was Napster, a music sharing system emerged in 1999. And after that resource sharing remains one of the most major applications of P2P overlay network. To some degree, we can also view service discovery as a type of resource sharing. As a synergy exists between the two types of networks, we argue that the methodologies used in P2P overlay network can also be applied to MANETs, and both the networks are faced with common fundamental challenge

Most of the structured P2P systems employ distributed hash table (DHT) to provide the resource discovery service. Each resource or service on the network is mapped to a key using some hashing algorithm, and each peer is responsible for a certain range of keys. This mechanism, which is called as consistent hashing tents to balance the load, each peer on the network takes responsibility for roughly same amount of keys. DHT enables an efficient resource discovery in P2P overlay networks, in particular, in an N-peer system, resource lookup can be resolved by delivering the query to the peer stores the corresponding information along a route of $O(log\,N)$ overlay hops in length with $O(log\,N)$ message exchanges involved.

By exploiting the synergy between two types of networks to design a effective service discovery protocol can be a promising research direction, however, simply integrate the conventional DHT with MANET is not an approach feasible enough. Although MANETs and P2P networks share many similarities as we have discussed above, differences also exist and can not be ignored.

The primary challenge to deploying a P2P overlay on MANETs is the fact that P2P overlays are designed to be used in the wired Internet, whereas mobile nodes in MANETs communicate via wireless channel. As a DHT substrate requires each peer keeps track of a set of overlay neighbors, the high mobility of nodes plus low reliability and limited bandwidth of the wireless link in MANETs makes the maintenance of consistency of prohibitively high cost.

The communication between nodes in MANETs is realized by the means of broadcast, which differs greatly from the behaviors of the wired networks such as LAN and Internet. As a result, when one node sends a packet, all the nodes within its transmission range will receive a copy and forward or simply discard it according to the contents carried by the packet. Therefore, if currently unneeded packets are overheard and cached before discarded, we can reduce the network traffic and augment the efficiency of routing and service discovery.

Another method can help to improve the performance is to group the mobile nodes into subnetworks, with the objective of maintaining a relatively stable topology and limiting the amount packets propagates in the network. A common way of grouping nodes into subnetworks dynamically is clustering, by which mobile nodes are assigned difference roles (cluster heads, gateways, etc.), and clusters are formed based on physical proximity. However, as nodes move freely and constantly in MANETs, complicated clustering protocol may introduce excessive network overhead with continuous cluster rearrangements, and outweigh the benefits it brings about. As a result, the clustering protocol must be designed carefully to resolve this trade-off.

Therefore, our envisioned service discovery protocol comprises a lightweight DHT overlay to locate the service and provide a complement to the routing table, an overhearing mechanism and a minimalist on-demand clustering protocol. To support the overlay, a reactive routing protocol is adopted to save precious network bandwidth.

In the next section, we present the related works and background about the study of service discovery in mobile ad hoc networks, distributed hashing table in both P2P and MANETs, and different clustering protocol.

In Section III, we introduce our proposed service discovery protocol, and mainly discuss the DHT-based service publish and service discovery mechanisms.

Section IV gives an optimization for the service discovery protocol. Using a Passive Clustering protocol[3], we construct a cluster structure and reduce the unnecessary flooding overhead without additional control packet costs.

In Section V, we present our simulation setup and results. And at the end of the paper, we conclude our work.

2 Background and Related Works

2.1 Service Discovery Models

Intensive research efforts have been dedicated in the field of service discovery. There are many service discovery protocol proposed for different application scenarios. Jini[4] is an architecture extending Java technology to enable construction of distributed systems consisting of services and clients. It provides service publish and lookup functionalities to allow the resources to be used across the network. Universal Service Discovery and Integration (UDDI)[2] is a standard designed to provide access to the web services registered in the web service directory. Nonetheless, these endeavors dwell on the deployment on mobile ad hoc networks. UDDI protocol, for example, employs publicly operated UDDI node or broker to provide service discovery. Service providers advertise the services to the broker, and clients obtain the service description by contacting the broker. As the availability of the services depends on the central directory, these approaches are not suited for a decentralized environment such as MANET as the central directory will overload the mobile device and become the bottleneck of the whole network.

Thence, decentralize service discovery protocols emerge to better suit the requirements of MANETs. Konark[5] and Allia[6] are typical service discovery protocols designed to work on MANETs. Allia employs a peer-to-peer cache mechanism to support the service discovery. Service providers advertise their services in the vicinity and the network is partitioned into proximity-based units call *alliances*. When a service discovery is issued, the requested service will be looked up with the *alliance*, upon failure, a broadcast or multicast of the request will be used.

Konark is a service discovery middleware in which nodes both actively discovery and passively cache service advertisements. A tree-structure-based service registry is maintained to manage the services.

[7] proposed another decentralized service discovery protocol by analogy of field theory. Service robustness is concerned and measured in this approach to elevate the reliability of the service.

2.2 Distributed Hashing Table

Distributed Hashing Table is a technology widely used in peer-to-peer applications to provide a lookup service. There are currently many highly structured DHT algorithm, such as Chord[8], Content Addressable Network (CAN)[9], Pastry[10], Tapestry[11] and Kademlia[12]. In the different implementations of DHT, every resource and peer in the P2P overlay network is assigned a unique key by a hash algorithm (e.g. SHA-1). The keys assigned, both to the resources and the peers, are distributed evenly on a same key space. The keys are mapped to peers using consistent hashing. Consistent hashing is a technique to deal with the change in the number of slots occurs in the system. Using consistent hashing, the remapping of keys to slots (peers in P2P scenarios) is reduced, and only K/n keys need to be remapped on average, where K is the number of keys and n is the number of slots. To be more specifically, for example, in Chord, all keys are arranged on a circular key space, a key k is owned by a peer whose key is closest to k on the circle out of all the other peers. When a peer joins or leaves the network, which consequently leads to a change in the number of slots, only the adjacent keys need to be rearranged. This feature is important for the distributed systems especially MANET, as the remapping of keys requires data exchange among the node involved, which can be quite expensive if too many nodes participate the remapping process.

Pastry and Tapestry are two similar DHT systems based on a technique called prefix-oriented routing[13] which is invented by Plaxton et al. before P2P systems came out. The idea of prefix-oriented routing is simple, every key of resource are always mapped to the node whose key has the longest common prefix with the resource's key. By using this technique, in a N-peer system, uniform data dissemination is achieved and routing can be finished in $O(log\, N)$ steps.

While it is not easy to apply a P2P application overlay to MANETs because of the high mobility and limitation in processing capacity, battery and bandwidth, there are already many attempts. Etka[14] proposed a cross-layer design to integrate a DHT (Pastry) with Dynamic Source Routing (DSR)[15] protocol. The efficiency of this DHT substrate is examined, which demonstrates a DHT substrate can be more efficient in supporting applications than a physical layer broadcast-based protocol. MADPastry[16] is another DHT substrate designed for MANETs. Unlike Etka, MADPastry employs ad hoc on-demand vector (AODV)[17] as its routing protocol to support a simplified Pastry substrate to better suit the environment of MANETs. In order to provide an explicit consideration of locality, MADPastry uses *landmark-keys* to evenly divides the key space. Nodes with keys closest to the landmark keys are viewed as *landmark-nodes* and send out beacons periodically in inform the presence in the vicinity. Nodes receiving beacon become cluster members of the landmark node. However, as the Pastry routing table is strapped down, the scalability of the system is sacrificed.

[18] proposed a more complex scheme. The authors aim to provides a DHT-based overlay for peer-to-peer Massively Multi-player Online Games (MMOGs) over MANETS. The concept of Random Landmarking [19] is used to create clusters and the clusters form a hierarchical tree structure which similar to the Hierarchical State Routing (HSR) protocol proposed in [20]. Each node maintains two types of DHT tables, the remote DHT table stores the landmark keys

of each cluster and the local DHT table is organized as a Virtual Ring Routing [21] table to support intracluster lookup.

2.3 Clustering Protocol

In general, clustering protocols can be classified into two types, proactive clustering protocols and reactive clustering protocols. A proactive clustering protocol attempts to build and maintain the clusters continuously. [22][23] present some proactive clustering mechanisms. These protocols focus mainly on the election of cluster heads and the behaviors of nodes joining or leaving a cluster. Control packets are transmitted even if there is no necessity to main the cluster structure, which is a waste of the limited bandwidth.

As for the reactive clustering protocols, [24] proposed a demand-driven clustering mechanism. A integration of the neighborhood recognition protocol and the clustering protocol is employed. Clustering setup is carried out only when there are packets to be sent, and the cluster signaling traffic is kept as minimal as possible.

The Passive Clustering protocol proposed in [3] is another reactive clustering protocol. The clusters are used to achieve efficient flooding in MANETs. All the cluster state information is piggybacked in the user data packets and no additional control packets are needed. The cluster infrastructure can be constructed as a byproduct of user traffic. Our proposed approach is largely inspired by this protocol.

3 Proposed Service Discovery Protocol

3.1 Overview

In this work, as our main focus is on the design of the service discovery protocol, we will concentrate on the behavior of nodes while in the process of service discovery and how they interact with lower layers, and evaluate the performance. Whereas we will not pay attention on the format of service attributes and types and how they are defined particularly. We adopt the definitions given in [25] about how service and service discovery should be understood throughout the paper:

- *Service* is any hardware or software feature that can be utilized by a mobile node.
- *Server* is any mobile node in the network that provides at least one service to the other nodes.
- *Client* is any mobile node that wants to utilize a specific service offered in the MANET.
- *Service discovery* is a mapping from a service description to the IP address of the server.

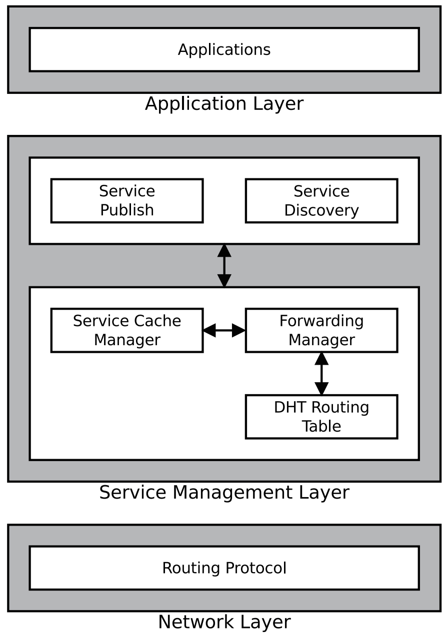


Fig. 1. Service discovery architecture

Figure 1 shows the components of our proposed service discovery architecture, which is composed of 3 layers: the application layer, the service management layer and the network layer.

The application layer is where the applications work. It provides the users of applications such as printing service, network storage, etc.

The core functionalities of the service discovery protocol are implemented on the service management layer. The servers publish the service into the network by invoking Service Publish module, and the Service Discovery module is used when the node acts as a client to lookup the service requested by the user. A service cache is maintain by each node to save the existence of and the service information about the neighboring nodes. When a node needs to send or forward packets, for instance a service discovery is initiated by the user, the DHT overlay routing table is used to help to find the next hop.

We use the well-known Ad hoc On-demand Distance Vector (AODV) as the underlying routing protocol to support the service discovery system. As AODV is a reactive routing protocol, lower traffic will be used to maintain the routing table.

3.2 DHT Based Service Discovery Protocol

In our proposed protocol, a Tapestry-like DHT overlay is used to support service publish and discovery. The DHT overlay is used to provide a guide to the nodes about where to find the request service.

In the network, each service provided by the service has it service description (SD), and every SD generates an unique key K_S by using a hashing algorithm (e.g. SHA-1).

Also, every node in the network has an unique node ID N_{id}, and from which an unique node key K_N is generated using the same hashing algorithm as what is used to get K_S. Because the hashing algorithms used are the same, the outputs, K_S and K_N are in the same key space.

Two basic functionalities provided by the service discovery protocols are as follows:

1. PublishService(SD, N_{id}): Publish service SD and make it available to the network. The server of the published service is identified by the node ID N_{id}.
2. DiscoverService(SD, N_{id}): Initiate a service discovery from the client N_{id} to get the information (IP address, hop count, etc.) of server which hosts service SD.

Publish Service Similar to the behaviors in Tapestry, every service in the network will have its key K_S mapped to a live node, which is called K_S's *root*. Among all the nodes, the root node's key K_N should be the closest to K_S. As there is no global knowledge about all the nodes in a distributed environment like MANET, the root node can not be selected directly from the key space, so the selection of root node is accomplished gradually in the process of service publish.

When a server needs to publish a service, it initiates a Service Publish packet containing the server's IP address and the service's key K_S. The node forwarding this packet will tentatively choose a destination node whose key K_N is closer to K_S than itself. In practice, that'll be a node whose key matching greater number of trailing bit positions will be chosen.

Figure 2 presents a synopsis for the service publish process. The server ($K_N = 1320$) needs to publish a service ($K_S = 2210$). Using the prefix-based routing, every node tends to send the packet to a node whose K_N has a longer common prefix with K_S (2210). Therefore, as is shown in Figure 2, the path $1320 \Rightarrow \underline{2}112 \Rightarrow \underline{22}33 \Rightarrow \underline{221}3 \Rightarrow \underline{2210}$ is adopted until the packet reaches the root, in which the underlined digits are the common prefix with K_S.

This routing strategy is similar to Class Inter-Domain Routing[26], but unlike the structure of subnetworks, the keys of the nodes have no proximity meanings. As can be observed in Figure 2, even the keys of intermediate node are gradually getting closer to the key of the root, the actual distance to the root is not always becoming smaller, but the overall direction is still towards the root. Another detail should be noticed is that each overlay hop is sometimes comprised of several physical hops in the actual network.

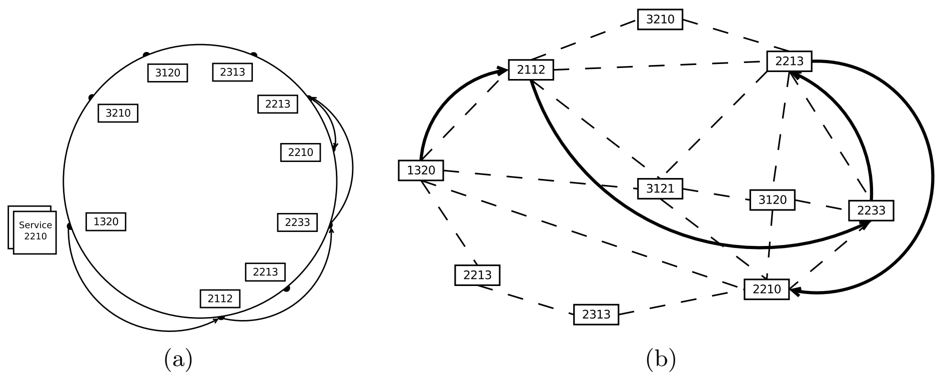


Fig. 2. Service publish example. (a) shows the path the packet would take on the key space ring, while (b) illustrates the path on the actual network topology.

In order to support the prefix-based routing, a DHT routing table should be maintain by each node. The number of columns and rows are determined by the bit length of the output of the used hashing algorithm, and the base chosen for keys. If the bit length is L and the base is 2^β, the DHT routing table will have $\lceil L/\beta \rceil$ rows with 2^β entries each. In the previous example of service publish process, the base 2^β chosen is 4 with $\beta = 2$, and the bit length L is 8. The entry at the ith row and the jth column is a pointer header linked to a list of information of nodes whose keys share the present node's key in the first $i-1$ digits, and the ith digit is j.

With the help of the DHT routing table, a node can easily pick out the destination node it should forward the SERVICE PUBLISH packet to. Here we define $lcp(K_1, K_2)$ to be the length of longest common prefix of K_1 and K_2, and K^i is the ith digit of K. The entry at $lcp(K_N, K_S)+1$th row and $K_S^{lcp(K_N,K_S)+1}$th column will link to a list of nodes which are closer to K_S than the present node. In certain cases, it's possible that the appropriate entry in the table is empty, the packet will be forwarded to a node sharing a prefix with K_S at least as long as the present node. This routing process terminates when no there is no appropriate entry available in the DHT table, and the present node will be viewed as the root node of this service. The service information, including the server's IP address and service description are stored in the root node.

Discover Service The service discovery process is similar to service publish. A SERVICE REQUEST packet is generated with the required K_S and the client's IP address included. Using DHT routing table, the packet is forwarded to the root node of the service in a similar hop by hop manner. As a service cache mechanism is employed, it is possible that an intermediate node on the packet forwarding path can provide the required service information, In that case, the

Service Request packet will not be forwarded to the next hop, instead a Service Reply will be initiated and send back to the client.

As the distributed manner of MANET, there is no global knowledge exists in one single node, as a result, there is possibility that the DHT routing table cannot provide valid information to tell the present node to which node the packet should be sent to. We adopt a best effort strategy here. In such cases, the present node start a flooding of Service Request to get the service information. In order to reduce the flooding packets, we use the following Passive Clustering protocol.

4 Passive Clustering Protocol

As discussed previously, our service discovery protocol uses AODV as the routing protocol. AODV is an on-demand routing protocol, which means there is no prior knowledge of the routing information to guide the sending of packets without a routing discovery. When the node starts the routing discovery, the routing discovery packet will be flooding to part of or the entire network. In addition, the service discovery process involves flooding as well when the DHT routing table cannot provide useful information.

Flooding consumes lots of bandwidth of network, and may cause broadcast storms when a large amount of nodes start flooding in a short period of time. However, if the network is dense and the transmission ranges of the nodes overlap a lot, many of the packet relays will be unnecessary. Therefore, we use Passive Clustering protocol[3] as an optimization for our protocol to reduce the unnecessary flooding relays.

Passive Clustering protocol assigns different roles to the nodes in the network, and partitions the network into dynamic clusters. The role of a node in the network can be ClusterHead, Gateway, Ordinary, and Initial according to the behavior of itself and the information collected from its neighboring nodes.

Initial serves as the initial state of the nodes. Nodes newly join the network are assigned Initial, and after a node stays inactive, which means no incoming and no outgoing network traffic for a period of time, it will revert to Initial.

Only Initial nodes are capable to become ClusterHead. When an Initial node has incoming traffic, it change itself to a special ClusterHead-Ready state. As Passive Clustering protocol involves no additional signaling traffic, the node cannot announce itself to be a ClusterHead node until it has packets to send, therefore the ClusterHead announcement is postponed to the time when a packet is to be sent.

A node hearing from two or more ClusterHead nodes becomes a Gateway node, as it will act as gateway between the two clusters. A node which is not a ClusterHead nor a Gateway is an Ordinary node. Normally, a node can only hear from one ClusterHead is an Ordinary node. Moreover, a Gateway hearing from more Gateway nodes than from ClusterHead nodes will revert itself to an Ordinary node.

The role of the node is piggybacked in the MAC header of the outgoing packets, which can be received or overheard by its neighboring nodes. These roles are not static, instead they are stored in soft state to better suit the dynamic topology. Timers are employed by the nodes to monitor the inactive time, and revert the nodes to INITIAL state if necessary.

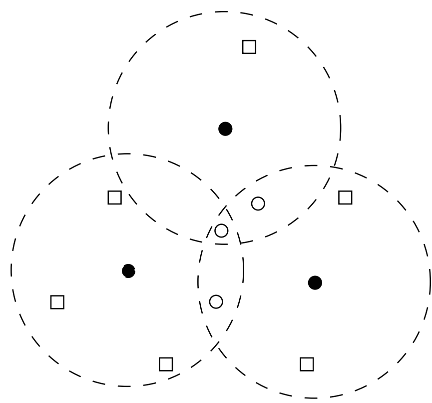


Fig. 3. An illustration of the clusters

Figure 3 shows a simple illustration of the cluster structures in a sample network. The three solid dots are three CLUSTERHEAD nodes. The dashed circles are the transmission range of the three CLUSTERHEAD nodes, which also define the ranges of the three clusters. The white dots in the intersection areas of two clusters represent GATEWAY nodes, and the remaining squares are ORDINARY nodes. The ORDINARY nodes are set to discard the received broadcast packets, and do not participate in the floodings. As a result, Passive Clustering protocol constructs and maintains a cluster structure in the network, and reduce the unnecessary flooding relay without introducing additional control packets.

5 Simulations

5.1 Simulation Setup

In the simulation, we implement our service discovery protocol on the wireless network simulator GloMoSim[27]. Some important parameters chosen for our simulation are shown in Table 1.

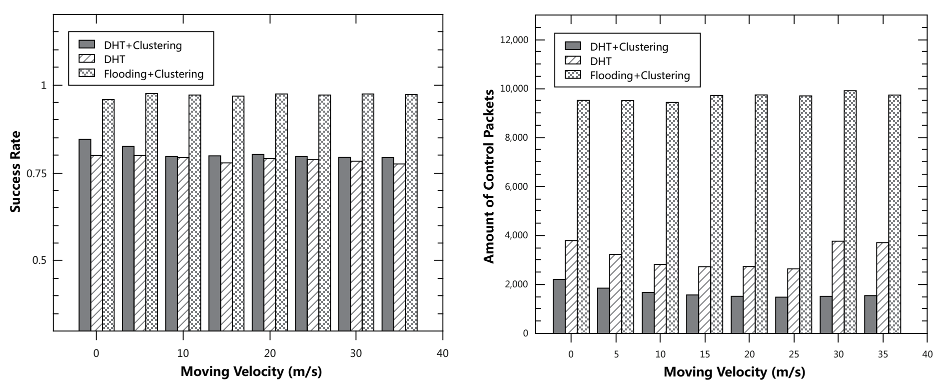
In this section, we evaluate the performance of our service discovery protocol by comparing the three difference implementations: the protocol with DHT and passive clustering enabled, the protocol with DHT but not using clustering and a flooding and passive clustering based service discovery protocol with service cache maintained by each node. These three approaches are marked respectively as *DHT+Clustering*, *DHT* and *Flooding+Clustering* in the figures below.

Table 1. Simulation Paremeters

Parameter	Value
Simulation Time	15 mins
Terrain Area	1000 m * 1000 m
Node Placement	random
Mobility Pattern	random waypoint

5.2 Simulation Results

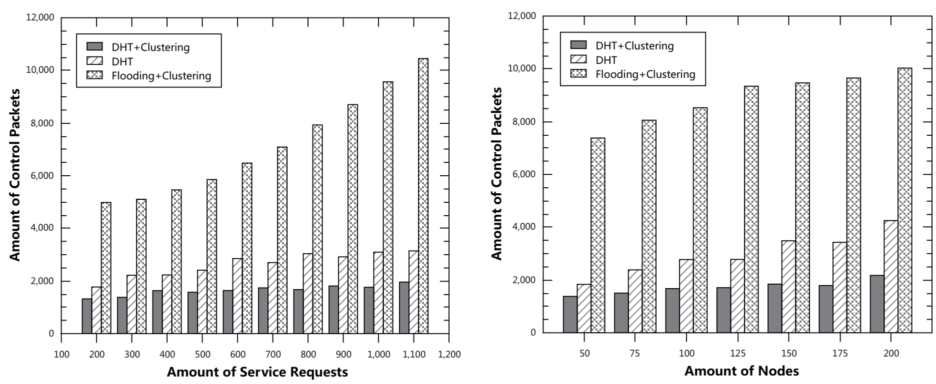
We first study the varying moving velocity's effect on the success rate of service discovery and the amount of control packets involved in the discovery process. The success rate is the percentage of successful service discovery among the total number of attempts. And the control packets consist of the SERVICE REQUEST packets, SERVICE REPLY packets, and the SERVICE PUBLISH packets. As the size of each type of packets are nearly the same, so we use the amount of control packets sent to the network to measure the pressure put on the network by the service discovery protocols.



(a) (b)

Fig. 4. The effects of nodes' moving velocity on the success rate of the service discovery and the amount of control packets used in the service discovery.

The Figure 4(a) shows that the *Flooding+Clustering* approach is a straightforward but feasible approach. As the client node broadcast a SERVICE REQUEST packet to the network upon each service discovery request, almost the entire network participate in the discovery process, the possibility that the service is discovered successfully is very high, almost all the service discovery attempts end successfully, and stable as well when the moving velocity varies. But what accompanies the high success rate is the huge amount of control packets as is shown in Figure 4(b). As our DHT based service discovery protocol makes a best effort attempt to deliver the Service Request to the server, the success rate of the two DHT based protocols are relatively lower than the flooding based one, but still tolerable. And this drawback can be overcome by retrying the service discovery when an attempt fails. Little network traffic will be introduced by the retries as the amount of control packets used in *DHT+Clustering* is very low as is shown in Figure 4. Also, the amount of control packets reduced by the passive clustering strategy can be observed in Figure 4(b).



(a) (b)

Fig. 5. The effects of number of nodes on the success rate of the service discovery and the amount of control packets used in the service discovery.

We also study the influences from other factors on the amount the control packets introduced by the service discovery. The results are presented in Figure 5. Figure 5(a) shows that the amount of service discovery request has an influence on the amount of control packets of all the three protocols, as more requests means more Service Request packets need to be sent. We can also observe that the rise in *Flooding+Clustering* is much more obvious than the other two. As can be observed in Figure 5(b), the amount of control packets also has a relationship with the amount of nodes in *Flooding+Clustering* and *DHT*. This is because more nodes will be involved in the forward of broadcasted packets.

In Figure 6 we have an analysis of the composition of the control packets used in the three types of protocols. In which we can observe the effect of clustering to reduce the flooding packets is significant, as the amount of Service Request packets broadcasted in *DHT+Clustering* is much lower than in *DHT*. And the effect of DHT can also be seen by comparing to *Flooding+Clustering*.

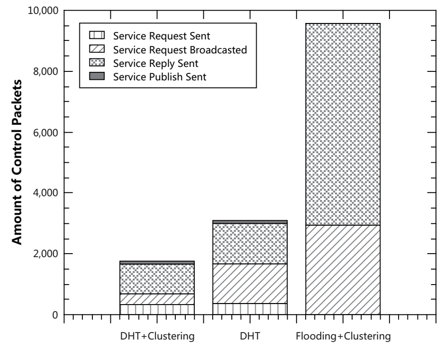


Fig. 6. The composition of the control packets used in the service discovery.

6 Conclusion

In this paper, we proposed a protocol aiming to provide a service discovery support to mobile ad hoc network. Our protocol is based on the idea of distributed hash table which is widely used in peer-to-peer networks and combines it with passive clustering to reduce the control packets. The protocol adopts a best effort strategy to decrease the complexity. The simulation results show our protocol is lightweight and has a relative sound performance.

References

1. : OASIS web services dynamic discovery (WS-Discovery) version 1.1. http://docs.oasis-open.org/ws-dd/discovery/1.1/os/wsdd-discovery-1.1-spec-os.html
2. : Standards | OASIS. http://www.oasis-open.org/standards#uddiv3.0.2
3. Kwon, T.J., Gerla, M., Varma, V., Barton, M., Hsing, T.: Efficient flooding with passive clustering - an overhead-free selective forward mechanism for ad hoc/sensor networks. Proceedings of the IEEE **91**(8) (August 2003) 1210–1220
4. Arnold, K., Scheifler, R., Waldo, J., O'Sullivan, B., Wollrath, A.: Jini Specification. Addison-Wesley Longman Publishing Co., Inc. (1999)
5. Helal, S., Desai, N., Verma, V., Lee, C.: Konark-a service discovery and delivery protocol for ad-hoc networks. In: Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE. Volume 3. (2003) 21072113
6. Ratsimor, O., Chakraborty, D., Joshi, A., Finin, T.: Allia: Alliance-based service discovery for ad-hoc environments. In: Proceedings of the 2nd international workshop on Mobile commerce. (2002) 19
7. Zhou, X., Ge, Y., Chen, X., Jing, Y., Sun, W.: Smf: A novel lightweight reliable service discovery approach in manet. In: Wireless Communications, Networking and Mobile Computing (WiCOM), 2011 7th International Conference on, IEEE (2011) 1–5
8. Stoica, I., Morris, R., Karger, D., Kaashoek, M., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. ACM SIGCOMM Computer Communication Review **31**(4) (2001) 149160

9. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: A scalable content-addressable network. ACM SIGCOMM Computer Communication Review **31**(4) (2001) 161–172
10. Rowstron, A., Druschel, P.: Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In: Middleware 2001, Springer (2001) 329–350
11. Zhao, B., Kubiatowicz, J., Joseph, A., et al.: Tapestry: An infrastructure for fault-tolerant wide-area location and routing. (2001)
12. Maymounkov, P., Mazieres, D.: Kademlia: A peer-to-peer information system based on the xor metric. Peer-to-Peer Systems (2002) 53–65
13. Plaxton, C., Rajaraman, R., Richa, A.: Accessing nearby copies of replicated objects in a distributed environment. Theory of Computing Systems **32**(3) (1999) 241–280
14. Pucha, H., Das, S., Hu, Y.: Ekta: an efficient dht substrate for distributed applications in mobile ad hoc networks. In: Mobile Computing Systems and Applications, 2004. WMCSA 2004. Sixth IEEE Workshop on. (dec. 2004) 163 – 173
15. Johnson, D., Maltz, D.: Dynamic source routing in ad hoc wireless networks. Mobile computing (1996) 153–181
16. Zahn, T., Schiller, J.: MADPastry: a DHT substrate for practicably sized MANETs. In: Proc. of ASWN. (2005)
17. Das, S., Perkins, C., Royer, E.: Ad hoc on demand distance vector (aodv) routing. Mobile Ad-hoc Network (MANET) Working Group, IETF (2002)
18. Yu, A., Vuong, S.: A DHT-based hierarchical overlay for Peer-to-Peer MMOGs over MANETs. In: Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International. (2011) 14751480
19. Winter, R., Zahn, T., Schiller, J.: Random landmarking in mobile, topology-aware peer-to-peer networks. In: Distributed Computing Systems, 2004. FTDCS 2004. Proceedings. 10th IEEE International Workshop on Future Trends of. (2004) 319324
20. Pei, G., Gerla, M., Hong, X., Chiang, C.C.: A wireless hierarchical routing protocol with group mobility. In: Wireless Communications and Networking Conference, 1999. WCNC. 1999 IEEE. (1999) 15381542
21. Caesar, M., Castro, M., Nightingale, E., O'Shea, G., Rowstron, A.: Virtual ring routing: network routing inspired by dhts. In: ACM SIGCOMM Computer Communication Review. Volume 36., ACM (2006) 351–362
22. Lin, C., Gerla, M.: Adaptive clustering for mobile wireless networks. Selected Areas in Communications, IEEE Journal on **15**(7) (1997) 1265–1275
23. Basagni, S.: Distributed clustering for ad hoc networks. In: Parallel Architectures, Algorithms, and Networks, 1999.(I-SPAN'99) Proceedings. Fourth International-Symposium on. (1999) 310315
24. Cramer, C., Stanze, O., Weniger, K., Zitterbart, M.: Demand-driven clustering in manets. In: International Workshop on Mobile Ad Hoc Networks and Interoperability Issues (MANETII04), Las Vegas, USA. (2004)
25. Kozat, U.C., Tassiulas, L.: Service discovery in mobile ad hoc networks: an overall perspective on architectural choices and network layer support issues. Ad Hoc Networks **2**(1) (2004) 2344
26. : RFC 4632 - classless inter-domain routing (CIDR): the internet address assignment and aggregation plan. http://tools.ietf.org/html/rfc4632
27. Zeng, X., Bagrodia, R., Gerla, M.: Glomosim: a library for parallel simulation of large-scale wireless networks. In: Parallel and Distributed Simulation, 1998. PADS 98. Proceedings. Twelfth Workshop on. (may 1998) 154 –161